# The MITLL NIST LRE 2015 Language Recognition System

*Pedro Torres-Carrasquillo, Najim Dehak*, Elizabeth Godoy, Douglas Reynolds, Fred Richardson, Stephen Shum*, Elliot Singer, and Doug Sturim*

Massachusetts Institute of Technology
Lincoln Laboratory
{ptorres,elizabeth.godoy,dar,frichard,es,sturim}@ll.mit.edu
*Computer Science and Artificial Intelligence Laboratory
{najim,sshum}@csail.mit.edu

## Abstract

In this paper we describe the most recent MIT Lincoln Laboratory language recognition system developed for the NIST 2015 Language Recognition Evaluation (LRE). The submission features a fusion of five core classifiers, with most systems developed in the context of an i-vector framework. The 2015 evaluation presented new paradigms. First, the evaluation included fixed training and open training tracks for the first time; second, language classification performance was measured across 6 language clusters using 20 language classes instead of an N-way language task; and third, performance was measured across a nominal 3-30 second range. Results are presented for the average performance across the 6 language clusters for both the fixed and open training tasks. On the 6-cluster metric the Lincoln system achieved average costs of 0.173 and 0.168 for the fixed and open tasks respectively.

## 1. Introduction and Task

The National Institute of Science and Technology (NIST) has conducted formal evaluations of language detection algorithms since 1994. In previous evaluations, NIST has explored issues related to language recognition ranging from closed-set language detection to confusable language pairs in the 2011 evaluation. In 2015 NIST pursued a different task and a new paradigm. The task for the NIST 2015 language recognition evaluation (LRE) was to determine the average performance of systems when classification within six predefined language clusters is considered. Additionally, the (mandatory) core condition for the 2015 campaign was a fixed training data task where all the data used for system development was provided by NIST. The evaluation also included a second optional condition where developers could construct their systems using any data that they had available.

The classification metric ***Cavg*** is defined as the average cost over the six clusters. As mentioned earlier, in contrast to previous evaluations, the 2015 LRE focused on classifying target classes within six language clusters. The language clusters included Arabic, Chinese, English, French, Slavic and Iberian. The breakdown of these language clusters is presented in Table 1.

Distribution A: Public Release.This work was sponsored by the Department of Defense under Air Force contract F19628-05-C-0002.

Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

$$Cavg = \frac{1}{NL}\left\{\left[C_{miss} * P_{target} * \sum_{L_T} P_{Miss}(L_T)\right] + \frac{1}{N_L - 1}\left[C_{FA} * (1 - P_{Target}) * \sum_{L_T}\sum_{L_N} P_{FA}(L_T, L_N)\right]\right\}$$

| Cluster | Target Classes |
|---|---|
| Arabic | Egyptian, Iraqi, Levantine, Maghrebi, Modern Standard |
| Chinese | Cantonese, Mandarin, Min, Wu |
| English | British, General American, Indian |
| French | West African, Haitian Creole |
| Slavic | Polish, Russian |
| Iberian | Caribbean Spanish, European Spanish, Latin American Spanish, Brazilian Portuguese |

***TABLE 1.*** *Language clusters for NIST LRE 2015.*

The overall performance measure, ***Cavg***, was computed for all submissions for both the fixed and open development tasks following the NIST LRE 2015 evaluation plan. [1]

The organization of this paper is as follows: Section 2 describes the partitioning of the data used for the MITLL submissions. Section 3 presents a description and the score fusion technique used on the submitted systems. Section 4 presents system performance on the NIST 2015 LRE task and a discussion of the results, with Section 5 presenting conclusions and suggestions for future work.

## 2. Development Data

The development data description covers two areas: data handing for the fixed condition and data used for the open condition. First, we will describe some of the commonalities covering both data sets and then discuss specific elements for each data set.

For both of these sets the Lincoln system used a common test set using the data provided NIST for the fixed condition. This test set consisted of segments generated by conducting speech activity detection on the files provided and extracting segments of duration no shorter than 3 seconds and no longer than 30 seconds. The duration of the files extracted was

uniformly distributed on the 3-30 seconds range to emulate the expected distribution of the evaluation segments. This test set covered roughly 40% of the total files provided by NIST for development. For languages where a large amount of data was provided and the duration of the provided files was longer than 5 minutes, the amount of files generated was limited to 10 files to reduce the possibility of biasing the performance of the system towards these languages. Table 2 describes the amount of data provided by NIST for each class.

| LANGUAGE | Cuts | Speech (hrs) |
|---|---|---|
| Iraqi (ara-acm) | 420 | 43.44 |
| Levantine (ara-apc) | 450 | 47.43 |
| Modern Standard (ara-arb) | 406 | 3.65 |
| Maghrebi (ara-ary) | 414 | 42.69 |
| Egyptian (ara-arz) | 440 | 97.27 |
| British English (eng-gbr) | 47 | 0.51 |
| Indian English (eng-sas) | 418 | 7.82 |
| American English (eng-usg) | 428 | 100.37 |
| Haitian Creole (fre-hat) | 323 | 2.51 |
| West African French (fre-waf) | 34 | 3.88 |
| Brazilian Portuguese (por-brz) | 47 | 0.75 |
| Polish (qsl-pol) | 487 | 31.25 |
| Russian (qsl-rus) | 470 | 18.44 |
| Caribbean Spanish (spa-car) | 120 | 29.84 |
| European Spanish (spa-eur) | 38 | 4.03 |
| Latin American Spanish (spa-lac) | 30 | 4.38 |
| Min (zho-cdo) | 41 | 5.27 |
| Mandarin (zho-cmn) | 438 | 74.62 |
| Wu (zho-wuu) | 45 | 5.07 |
| Cantonese (zho-yue) | 23 | 2.56 |

***TABLE 2.*** *Development data distribution as provided by NIST. The NIST language codes are in parentheses.*

### 2.1. Fixed Condition

For the fixed condition, the remaining 60% of the data provided by NIST was used for training. The data available for some of the languages was very limited as observed in Table 2. To help reduce the impact of this data limitation in our system, multiple data augmentation techniques were considered ranging from simply reusing the same data by using both full files and segments generated from these files (effectively letting the systems use the same data file twice) to modifying the speech signal via warping and tempo modification. The only technique that showed consistent gains on contrastive experiments for our systems was the data reuse into both full files and segments.

### 2.2. Open Condition

As described earlier, the open condition allowed for development of the systems using any data sources and amounts deemed necessary by the system developers. For this condition, data was used for development from sources including:

- Telephone data from previous LREs (2007, 2009, 2011), OHSU, OGI-22, Fisher, CallFriend, Babel, Ahumada, MI5-UK, Appen, Qatar-Dialect, and Kalaka
- Narrowband segments from VOA broadcasts.

sources of data were considered, during development it was observed that using the additional data hurt performance on our initial experiments. Additional experiments showed that judiciously adding data to some specific classes helped improve performance. This issue will be discussed in more detail in Section 4.

## 3. Classifiers

As in previous LREs, the Lincoln language recognition system consisted of the fusion of multiple classifiers. For LRE 2015, systems developed were largely based on the i-vector framework [8]. In this section, we describe the different classifiers and the fusion/calibration strategy.

### 3.1. Bottleneck features Classifiers

Eleven systems were considered, with ten of them based on the i-vector framework and with many of the systems using bottleneck features in some form.

#### *3.1.1. Bottleneck features*

The bottleneck features (BNF) used for the various systems are obtained by training a Deep Neural Network (DNN) using a seven hidden layer architecture. On these systems, all hidden layers have 1024 nodes except for the sixth layer which has either 64 or 80 nodes with a linear activation function that is used for extracting the BNFs. The output layers have varying compositions for the different systems and will be discussed for each system separately. Other features that were common across the systems include:

- Processing speech window of 20 ms length with 10 ms shift. Mean subtraction is performed and low energy dither added to the signal to avoid digital zeros.
- Mel-scale filterbank analysis is conducted over the band 0-4000 Hz resulting in 24 log-filterbank energies. RASTA filtering is applied to the log-energy filterbank trajectories.
- Non-speech frames are gated out using speech activity detection marks derived from a GMM-based speech/non-speech detector.
- Feature vectors are normalized to zero mean, unit variance by subtracting the mean and dividing by the standard deviation computed from either a 3 second window of speech frames or from the entire file.
- PLP features are computed with coefficients 0-12 kept for processing.
- In the cases where SDC features are used, cepstral coefficients 0-6 are used.

#### *3.1.2. Conventional bottleneck feature systems*

Two core bottleneck feature systems were developed following the architecture described above. The first bottleneck system, named **BNF1**, uses 1024 nodes in each hidden layer and a bottleneck layer of dimension 80. This DNN is trained using 90% of the Switchboard (SWB) phase 1 dataset. The training for this DNN uses the Kaldi toolkit [2] to extract 4168 senone posteriors. The feature set used to train

the network uses a stack of 21-frames of dimension 39 which includes 13 static cepstral coefficients plus both first and second derivatives. The bottleneck feature vectors obtained are normalized to follow a standard normal distribution and used to train a GMM-UBM [3] and subsequently generate a set of 400-dimensional i-vectors. Additionally, this system employed data augmentation techniques using the scheme proposed by Ko [4] The augmented data set was used to train the linear discriminant analysis (LDA) component and the within class covariance normalization (WCCN) matrix. Scores for this system were generated using cosine scoring.

A second BNF system (**BNF2**) was also trained using a scheme similar to **BNF1**. In the case of **BNF2**, the DNN was trained using a 100-hour subset of the SWB data set and the bottleneck dimension was 64. In this case, the system training resulted in the extraction of 4199 senone posteriors. The architecture and parameters are the same as **BNF1** with the exceptions described.

### 3.2. DNN posteriors systems

Another group of systems was trained using DNNs for direct computation of the sufficient statistics in lieu of using a GMM-UBM system. In this case, the systems use the DNN senone posteriors to compute sufficient statistics used to train the i-vector extractor. Under this general approach we considered four systems, all of which employ an i-vector framework and cosine distance scoring.

#### *3.2.1. Multinomial subspace systems*

Three of the systems evaluated (**CNT1**, **CNT2**, and **CNT3**) used the same framework as developed for the **BNF1** system with a difference in the DNN architecture. In this case, although 4168 senones were also used, the architecture of the system uses hidden layer dimensions alternating between 2048 and 1024 nodes. Posterior statistics are extracted for each hidden layer. Additionally, the subpace multinomial model is applied and an 800-dimensional space is ultimately used.

The first system (**CNT1**) modeled all 4168 posteriors while the second system (**CNT2**) modeled 20 posteriors representing the 20 classes of interest among the 6 language clusters. The third multinomial subspace system (**CNT3**) used DNN posteriors and language class posteriors jointly.

#### *3.2.2. Statistics based system*

This system (**STATS**) follows the description for **BNF2** but uses the 4199 senone posteriors along with the 56-dimensional shifted-delta-ceptral (SDC) features [5] to extract the first and second order statistics for i-vector extraction.

### 3.3. Bayesian Unit Discovery (BAUD)

The **BAUD** system is also a BNF system but it uses a different approach to determine the units by which the initial DNN targets are trained. In this case, instead of training the DNN using senone targets from the tri4a step of the Kaldi SWB recipe [2], this system trained its bottleneck features using targets from an unsupervised unit discovery process detailed below. The architecture for the DNN is the same as that for **BNF2**.

The unsupervised unit discovery process is based on the work in Lee [6], but was subsequently re-implemented in Kaldi with a few simplifications to make the computation more tractable [7] The main idea is to learn phone-like units on speech without parallel text data. Each unit is represented by a 3-state HMM that emits acoustic feature vectors via a GMM. In Lee [6], everything was formulated in a Bayesian manner to take advantage of its self-regularizing model-selection properties, and inference was done via Gibbs sampling. In the faster re-implementation, we used a more heuristic initialization, which included specifying the number of units to learn, and accumulated GMM statistics via maximum likelihood.

We learned 100 units on all of the provided training data. This resulted in a large set of "phone sequences" from which we could train a speech recognizer in Kaldi. Carrying through to the tri2 step of the SWB recipe resulted in an acoustic model containing 2604 senones modeled using 30,000 Gaussians. The frame-level alignments for these senones were used to train the DNN for bottleneck feature extraction.

### 3.4. Conventional SDC features system

One system was included that used conventional SDC features in an i-vector framework [8]. This system is similar to the system submitted in LRE 2011 and used an 7-1-3-7 SDC feature set along with static cepstra for a 56-dimensional vector. Processing of the speech signal is also described in [9].

### 3.5. Pitch features

Two systems that included pitch information were considered for this evaluation. The first pitch based system (**PITCH1**) used pitch stacked with SDC features using the system described in Section 3.4, and the second system (**PITCH2**) added pitch as input to the **BNF2** system. The pitch features were generated on a per-cut basis. Praat [10] was used to calculate F0 and the corresponding voicing decision using a 10 millisecond frame rate, and with the F0 range set to 65-400 Hz. To mitigate the effects of pitch doubling and pitch having, the highest and lowest 3% of F0 values were removed. The log of F0 was computed and its mean over the voiced frames of the cut was subtracted. Linear interpolation of log(F0) was performed through the unvoiced frames and those with the most extreme F0 values were removed. Delta-log(F0) was calculated as the difference between the log(F0) value 3 frames forward and 3 frames back in time. The values of log(F0) and delta-log(F0) were stacked with the corresponding SDC frames, producing a 58 dimensional feature vector for the **PITCH1** system. The **PITCH2** system used values of log(F0) and delta-log(F0) stacked with the **BNF2** system features.

### 3.6. Multi-lingual DNN

All the systems described in Sections 3.1-3.5 were developed in the context of the fixed condition. In addition to those systems we also developed an additional bottleneck system for the open condition. The multi-lingual DNN system (**MLBNF**) developed for the open condition was inspired by the work in [11], where a multi-task DNN was trained using data from 5 IARPA Babel languages (Cantonese, Pashto, Turkish, Tagalog, Vietnamese) as shown in Table 3. The DNN was trained used 60 hours of data randomly selected from each language for a total of 300 hours of data. The inputs for the DNN were the same stacked features used for the **BNF2**

system. The DNN architecture is also similar to the **BNF2** system in that it has 7 layers of 1024 nodes each where the second to last layer is a 64 node linear bottleneck. However, for the multi-lingual DNN the last hidden layer is different for each of the five languages. Stochastic gradient descent training for the multi-lingual DNN proceeds by loading a mini-batch with data from each language in sequence until the average validation cost across all languages no longer decreases.

| Language | IARPA Build Pack |
|---|---|
| Cantonese | IARPA-babel101b-v0.4c |
| Pashto | IARPA-babel104b-v0.bY |
| Turkish | IARPA-babel105b-v0.4 |
| Tagalog | IARPA-babel106b-v0.2g |
| Vietnamese | IARPA-babel107b-v0.7 |

*TABLE 3 Babel languages used for training a multi-lingual BNF.*

### 3.7. Fusion/Calibration

As in previous evaluations [9], the backend stage consisted of a per-system calibration component that included duration normalization. The calibration stage is then followed by a linear fusion with a zero offset. The calibration stage used a discriminatively-trained (MMI) Gaussian with shared covariance for each system, followed by a multiclass logistic regression across systems to produce the final score. With the limited amount of data in the evaluation, the submitted system was trained on a combination of the train and development scores.

To select the primary system we used a greedy approach to choose from among a maximum of 16 possible systems for the final system combination. System combinations were evaluated in three consecutive stages starting with a subset of the systems and then dropping and adding systems at each stage to select the best combination.

## 4. Results and Discussion

This section presents the results for the primary systems submitted for the 2015 NIST LRE, including both the open and fixed condition systems.

### 4.1. Official NIST Submission

Results for the fixed condition primary system are shown in Figure 1. The figure shows results for the primary system along with each of the individual systems that made it into the primary submission and the optimal (oracle) fusion. The submission consisted of a fusion of five systems: **BAUD**, **CNT1**, **BNF1**, **PITCH1** and **STATS** systems. This fusion was obtained by sweeping over all the systems developed for the evaluation and choosing the best compromise between performance and possible overtraining risk. The overall performance for the primary system had an average cost ***Cavg*** of 0.176. Figure 1 shows two performance bars: the performance of the submission across all six clusters (blue) and the performance of the system with the French cluster excluded. Additional comments on the performance of the system and the issues with the French cluster are included in Section 5.

There are a number of observations of interest to be made about these results. First, the submission choice was very close to the optimal performance possible given the developed systems. This result demonstrates that although the development set was not very good about predicting performance on the evaluation, the fusion strategy was a good predictor of which systems to combine. Second, the **BNF1** system results in the best performance out of the five systems submitted with the **BAUD** system resulting in a close second.

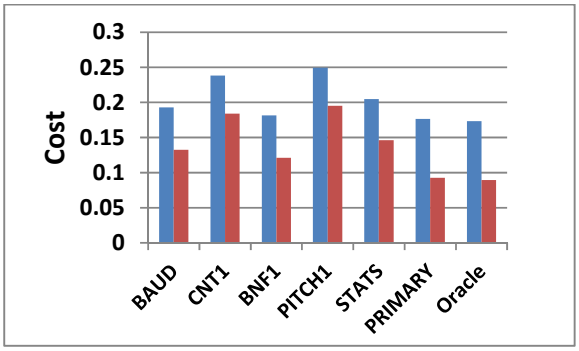


*FIGURE 1. Fixed task performance breakout.*

The open set condition submission performance breakout is shown in Figure 2, once again with average performance across all six clusters shown in blue with the performance with the French cluster excluded presented in red. For this condition, candidates included all systems considered for the fixed task along with the multi-lingual DNN bottleneck features system **MLBNF**. The performance for our submission resulted in an actual average cost of 0.169. As in the case of the fixed condition, the performance of the primary submission is on par with the optimal possible combination of systems. Another observation of interest is that for the open task submission the multilingual BNF system **MLBNF** is the best performing system and replaces the **BAUD** system.

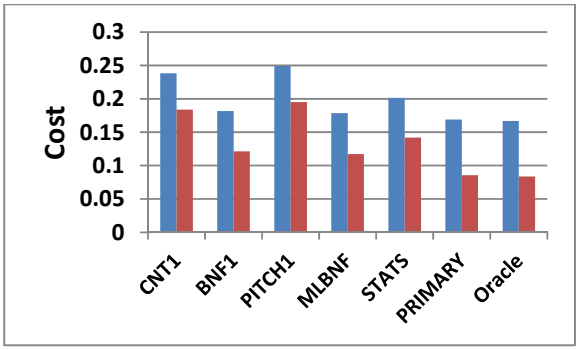


*FIGURE 2. Open task system performance breakout.*

## 5. Discussion

In this section we discuss some of the results obtained for the evaluation along with some of our observations and lessons learned. Topics include system development, duration

analysis, performance on the French cluster, and our experience with the open set condition.

### 5.1. Development Results

First, we describe our development results to motivate some of the decisions discussed earlier. Figure 3 presents the results obtained on our development set for both the fixed and open set conditions.

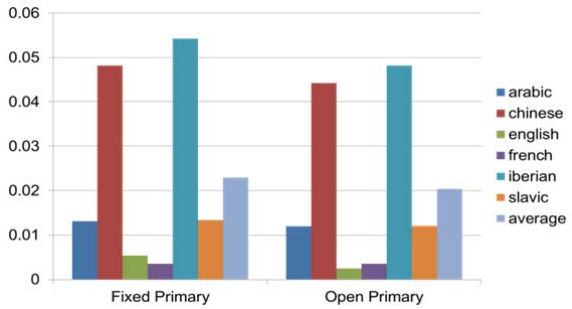


*FIGURE 3. Fixed and open system results on development data.*

The results demonstrate that the system development process predicted good performance for all six clusters with English expected to be the cluster with the easiest discrimination task and Chinese and Iberian expected to be the hardest discrimination tasks. This result contrasts with those obtained by our systems on the evaluation data where French was the worst performing cluster followed by the Iberian and Chinese cluster. Our analysis to date suggests that most of the differences are related to unexpected channel mismatch compared to the development set including among limited representation of the evaluation channels on the development data for the most difficult clusters.

### 5.2. French cluster performance

Figure 4 shows the performance of our fixed primary submission system across each of the language clusters, French being of particular interest as the performance of our system is very poor and well below expectations. As shown in Table 1, the French cluster was composed of Haitian Creole and West African French. Anecdotally, the performance on this task was expected to be difficult but not random.

Upon further investigation, we discovered that one of the main issues driving the performance degradation on the French cluster was the channel differences. During our system development process the data available for these classes had limited cross channel representation while the data used on the evaluation resulted in a large cross channel testing scenario. To further clarify this point the data for the French cluster is projected into two dimensions in Figure 5,

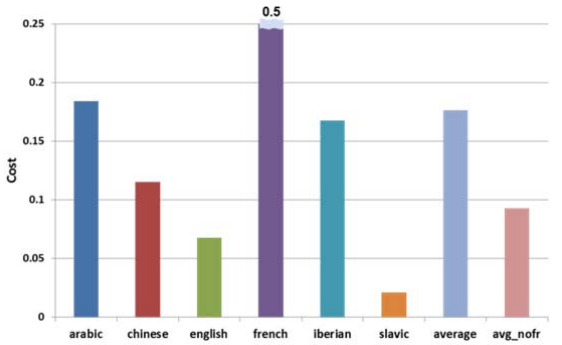


*FIGURE 4. Per language cluster performance for fixed set condition.*

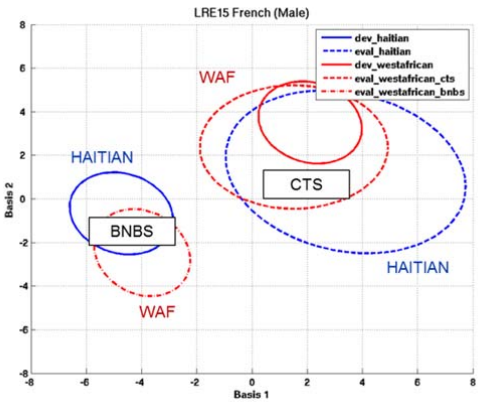


*FIGURE 5. French male data cluster projection into two-dimensional space.*

which demonstrates that there is likely a strong mismatch between the available training/development data and the evaluation data. In the figure the data that comes from conversational telephone speech (CTS) clusters together with the data originating from broadcast news (BNBS) sources also clustering together. Looking at the CTS and BNBS clusters it is clear that very limited language discriminability is to be expected.

In contrast, we show in Figure 6 two-dimensional projections for the Slavic cluster, which was selected because it comprised two language classes that were readily distinguished. We can observe a very different situation to that of Figure 5. In this case, we can observe a very clear differentiation of the two languages (Polish and Russian) and see that the clusters can be separated by channel but can also be separated by the language class.

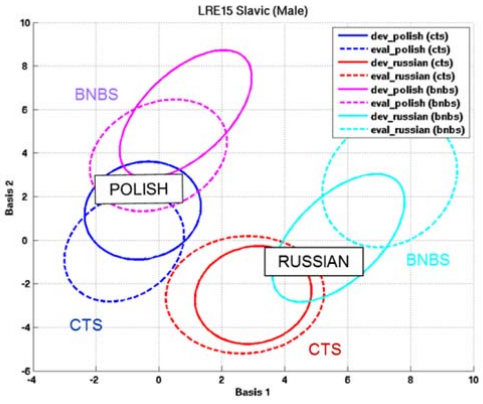


*FIGURE 6. Slavic male data cluster projection into two-dimensional space.*

### 5.3. Impact of duration

In previous language evaluations NIST had explicitly included duration as part of the evaluation. In 2015 NIST did not include duration as a main factor to consider and provided the evaluation data as a single set with durations in the (nominal) 3-30 second range. Figure 7 presents the performance observed across the different clusters (French is excluded). It is worth observing that, as expected, performance on current systems improves as the duration of the cut increases, with the saturation area around 15 seconds for most clusters.

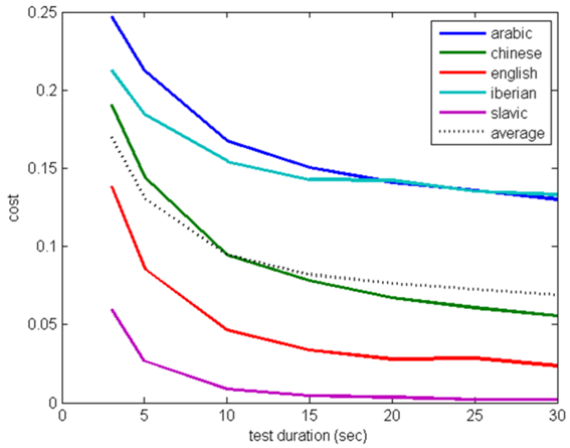


*FIGURE 7. Performance as a function of duration across all six clusters (including average performance).*

### 5.4. Open set system

The open set condition resembles the core condition of previous evaluations. One of the main observations from the development of the open condition systems in LRE 2015 was that adding additional data to the system training resulted in minor gains in performance compared to what had been our experience from previous evaluations. In fact adding all training data to our training resulted in some performance degradation. After additional experiments we tried to isolate the cases under which additional data produced performance improvements. Our experiments showed that adding data, one language at a time, improved performance for only the cases where data was added for Brazilian Portuguese, British English and Modern Standard Arabic.

After submission of system we revisited the open condition training on the evaluation data. Surprisingly, and in contrast to the development results, using all available training data does result in improved performance on the evaluation set. The average cost for the submitted system for the open condition was 0.168, while the post-evaluation system trained on all available data results in an average cost of 0.117.

In addition to the improved performance obtained on the post-eval system two other interesting observations are noted. First, contrary to the results obtained on the submitted system, performance on the French cluster improves substantially in the post-eval system. This improvement is possibly due to the additional diversity in channels available on the augmented data set. Although the French cluster accounts for most of the improvement, the performance on other clusters also improves. A second observation is that PLDA scoring performs better than both WCCN and conventional cosine scoring. This result also differs from that obtained during the system development phase on both the fixed and open conditions.

### 5.5. Overall system performance

Another analysis explored the performance of our system using a 20-way closed set classification metric rather than ***Cavg*** as for the 2015 NIST LRE. Figure 8 shows the performance of our primary system as a 20-way classification task. The axes show the 20 classes in the evaluation grouped by cluster.

The structure on the plot in Figure 8 shows that the confusions appear in roughly rectangular shapes about the diagonal. This is expected as the majority of confusions for this data happen within individual language clusters.

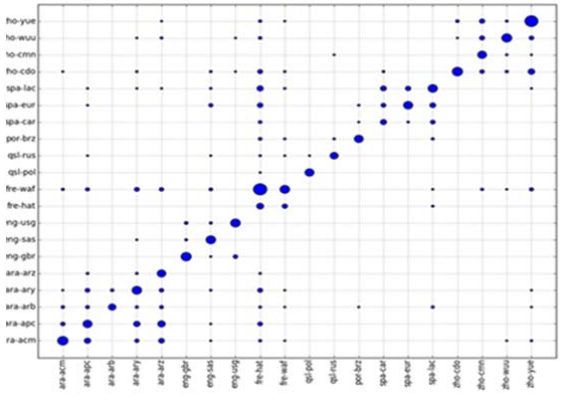

*FIGURE 8. Bubble plot of confusion matrix from MITLL system.*

### 5.6. Historical LID performance

As in previous evaluations, Figure 9 shows the historical performance trend for MITLL language identification submissions to NIST evaluations. The reader should keep in mind that the values shown demonstrate the performance of the systems using the technology at the time of submission and does not reflect the performance that could be obtained on this data with state-of-the-art systems.

The performance observed for the 2015 LRE is slightly higher than that obtained on recent evaluations. We hypothesize that the difference in performance can be due to the choice of target classes and the channel mismatch in some of the classes.

## 6. Conclusion

In this paper we have described the MITLL submission to the 2015 NIST Language Recognition Evaluation. MITLL submissions included both a fixed condition submission and an open set submission. The submissions were mainly based on systems using an i-vector framework and resulted in an average cost of 0.173 and 0.168 on the fixed and open tasks, respectively. In the future we intend to conduct additional analysis and listening to better understand the cluster confusions.

This evaluation relied heavily on systems based on Deep Neural Networks and bottleneck features. In the future we expect the issue of channel robustness to be central to performance and anticipate that future work will focus on using the new techniques that have recently emerged that exploit DNN approaches for channel compensation.

## 7. References


[1] 2015 Language Recognition Evaluation plan. http://www.nist.gov/itl/iad/mig/upload/LRE15_EvalPlan_v23.pdf

[2] D. Povey et al., *"The kaldi speech recognition toolkit,"* in Proc. IEEE ASRU, 2011.

[3] Reynolds, D. A., Quatieri, T. F., Dunn, R. B., *"Speaker Verification Using Adapted Gaussian Mixture Models"*, Digital Signal Processing Review Journal, January 2000.

[4] Tom Ko, Vijayaditya Peddinti, Daniel Povey and Sanjeev Khudanpur, *"Audio Augmentation for Speech Recognition"*, Proc. Interspeech, 2015.

[5] Torres-Carrasquillo, P. A., Singer, E., Kohler, M. A., Greene, R. J., Reynolds, D. A., and Deller Jr., J. R., *"Approaches to Language Identification Using Gaussian Mixture Models and Shifted Delta Cepstral Features"*, In Proc. International Conference on Spoken Language Processing in Denver, Colorado, ISCA, pp. 33–36, 82–92, September 2002.

[6] C. Lee and J. Glass, *"A Nonparametric Bayesian Approach to Acoustic Model Discovery,"* Proceedings of ACL, July 2012

[7] Thanks to David Harwath (SLS, MIT CSAIL) for providing the code and support of his re-implementation of BAUD in Kaldi

[8] N. Dehak, P. Torres-Carrasquillo, D. Reynolds, and R. Dehak, *"Language Recognition via Ivectors and Dimensionality Reduction,"* Proc. Interspeech, pp. 857-860, Florence, Italy, August, 2011

[9] E. Singer, P. Torres-Carrasquillo, D. Reynolds, A. McCree, F. Richardson, N. Dehak, and D. Sturim, *"The MITLL NIST LRE 2011 Language Recognition System,"* Proc. Odyssey, pp. 209-215, Singapore, June 2012.

[10] Paul Boersma & David Weenink (2013): *Praat: doing phonetics by computer* [Computer program]. Version 5.3.51, retrieved 2 June 2013 from http://www.praat.org

[11] R. Fer, P. Matejka, F. Grezl, O. Plchot and J. Cernock, *"Multilingual Bottleneck Features for Language Recognition,"* Proc. Interspeech, 2015

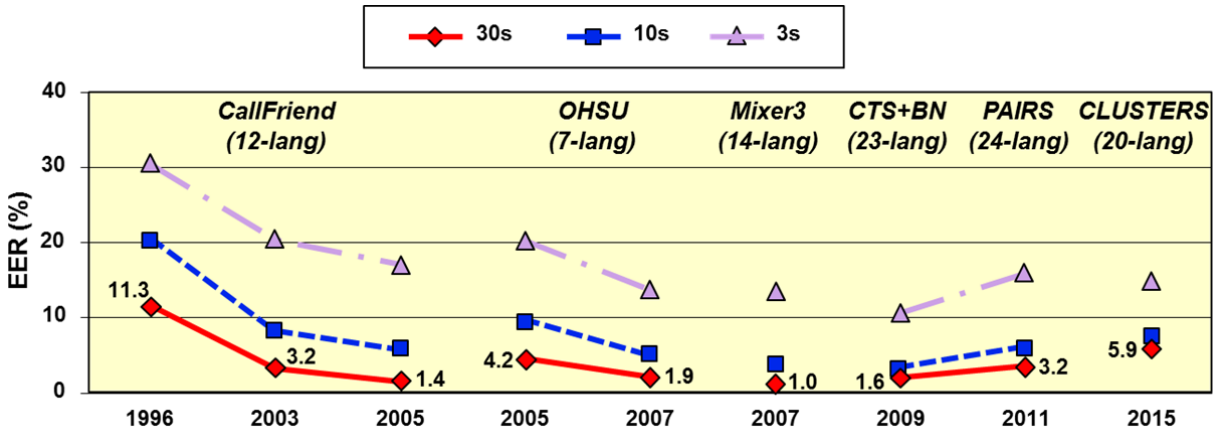


*Figure 9. Historical performance trend on NIST LREs from 1996-2015.*